Vol. II, No. 6
July/August 1983

SATN®

Software Arts Technical Notes from the creators of *VisiCalc*®

CONTENTS	FEATURING	PAGE
Round and Round We Go VisiCalc normally carries numbers to 11 or 12 significant digits. But what if you want values rounded, for example, to the nearest eighth or to the next lower 50 or to the next greater half? Here we discuss how to round values in whatever way you choose—up, down, or to the nearest specified interval.	@IF @INT @ROUND	2
Overcoming Value Limitations Using @LOG10 Even VisiCalc has its limits. On most machines it doesn't permit values larger than 1E62 or less than 1E – 66. It is unlikely that you run into these limits very often, but there is a solution when you do. In this article we demonstrate how to use @LOG10 to keep your values within allowable limits. In the process, we talk about the mathematics of logarithms, factorials, and using DIF files to reduce recalculation time.	@LOG10 @ERROR /S#S /S#L ^	4
Book Review Ron Klinger, Editor of *The Journal of Taxation*, reviews Stanley Trost's *Doing Business with VisiCalc.*		7
***Turning the Tables on Taxes* Revisited** Using the rounding technique developed in *Round and Round We Go,* we extend the utility of the model developed in the March/April 1983 issue of SATN to calculate Federal income tax. The modified model can be used both when the Tax Table is needed and when the Tax Rate Schedules apply. In future years, simply change the numbers for base tax and marginal tax rates to reflect new Federal figures and the model is ready to use.	@IF @INT @MAX	8
Related Products Checklist Once again we list some of the products that enhance the VisiCalc program.		10
Index to Volume II		11

Round and Round We Go

There are times when you don't want the result of a particular calculation, but rather that number rounded in a certain way. For example, you might want financial calculations made only to the closest penny, rather than retaining 11 or 12 significant digits as VisiCalc® normally does. In dealing with stock prices, you might want to round values to the nearest eighth. In inventory control you might need to round values up: If your calculations tell you that to produce a certain volume of goods you need 1.3 machines, you actually need two since you cannot have one-third of a machine. Similarly there might be cases where you want to round values down.

We're not talking about simply changing the *display* of values, but rather the values themselves. Formatting commands, such as /F$, change only the appearance of values while retaining the full 11 or 12 significant digits.

Using @INT, one of VisiCalc's built-in functions, rounding can be done automatically.

Rounding Positive Numbers

Rounding positive numbers to the nearest specified interval is the most common requirement. In its standard VisiCalc form it is:

*rounded number = @INT(original number * factor + .5)/factor*

@INT strips all digits to the right of the decimal point, returning the integer portion of the specified number. Thus, @INT(3.5) is 3, @INT(.785) is 0, @INT(−3.5) is −3, and so on. The factor in the above equation varies depending on the interval you want to round to. The added term, .5, "kicks up" the rounded value if the original number lies halfway or more along a given interval.

For example, to round a number to two decimal places (that is, to the nearest hundredth or the nearest penny) the factor is 100, so the expression is:

*rounded number = @INT(original number * 100 + .5)/100*

We'll set up the sheet so you'll enter the number you want to round in B1 and the interval in D1, as shown in Figure 1. Set the column width to 10 (/GC10) and the recalculation order to rows (/GOR).

After entering the appropriate labels, enter the following:

```
>B1:123.4567
>D1:100
>B4:@INT(B1*D1+.5)/D1
```

The value returned for B4 is 123.46, the nearest hundredth. Change the value of B1 to 123.4512. B4 changes to 123.45. The expression at B4 rounds to the nearest interval, up or down.

Change the factor to 10, meaning you want to round to the nearest tenth. B4 changes to 123.5. Change the factor to 1/10 to round to the nearest 10. B4 changes to 120. And so on.

The expression in B4 is in the standard form for rounding, but it is clumsy—to round to hundredths (that is, to the nearest .01) you use a factor of 100, while to round to the nearest hundred you use 0.01. It is more convenient to enter a value of 1/100 to round to the nearest hundredth and 100 to round to the nearest hundred. That is, we want to enter in D1 the reciprocal

How To Use SATN

The VisiCalc® program is available on many different personal computers. The keyboards on the machines vary slightly. To reduce confusion, SATN will use the symbol ® and the word RETURN synonymously to represent the keys known as ENTER, ENDLINE, and RETURN. The symbol ^ is used to represent the exponentiation operator.

The VisiCalc program was designed so that the colon (:) has the same effect as RETURN in some commands, as when ending the > command and to separate the various elements of the replicate command. SATN uses the same convention.

In examples of what to type, type exactly what you see and press RETURN at the end of every line. On some computers, you may find it necessary to add parentheses to some of the expressions included in the text. For example, 3+4*5 may become (3+4)*5.

Throughout SATN, one typeface represents both VisiCalc input and output (what you should type and what you will see on the screen). The typeface looks like this:

```
This line is typeset in OCR-B.
So is this: +A1+B1*3.25
```

To allow you to type the formulas directly into VisiCalc, the examples use specific coordinates, like A1, rather than the names of the value they represent. When actually typing a formula, you can use the arrows to point rather than using the > command.

SATN is published bimonthly by Software Arts Products Corp., 27 Mica Lane, Wellesley, MA 02181. Address all editorial correspondence to the Editorial Department, SATN, 27 Mica Lane, Wellesley, MA 02181. Requests for special permission or bulk orders should be addressed to the publisher.

Domestic subscriptions: 6 issues $30.00, 12 issues $50.00; outside U.S.A., 6 issues $50.00, 12 issues $90.00. Single copy price, $6.00. Send subscriptions, change of address, and fulfillment questions to SATN Subscriptions, P.O. Box 815, Quincy, MA 02169.

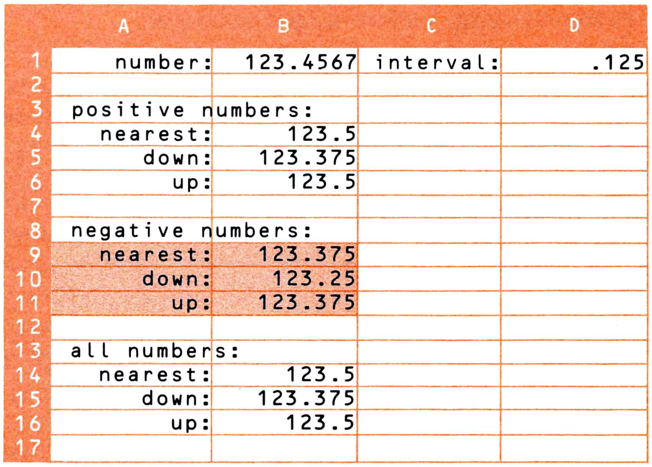
Copyright © 1983, Software Arts Products Corp. All rights reserved. This periodical may not be copied or reproduced in any fashion either in whole or in part without the prior written consent of Software Arts Products Corp.

Software Arts Products Corp. makes no warranty, express or implied, with respect to the quality, accuracy or freedom from error of the information contained in SATN, the journal for VisiCalc users, and specifically disclaims all liability for any damages resulting from the use of such information.

Unless specifically stated otherwise in any article in this issue of SATN, Software Arts Products Corp. has not examined any products and/or publications identified in SATN to determine their accuracy and/or ability to perform as advertised. Mention of a product and/or publication in SATN does not constitute an endorsement by Software Arts Products Corp. of any such product or publication.

Software Arts is a trademark of Software Arts, Inc. and Software Arts Products Corp. Unless specifically stated otherwise, all references in this issue of SATN to Software Arts shall refer to both companies. SATN is a registered trademark of Software Arts Products Corp. DIF is a trademark of Software Arts Products Corp. VisiCalc and VisiPlot are registered trademarks of VisiCorp.

Apple is a registered trademark of Apple Computer, Inc. IBM is a registered trademark of International Business Machines, Inc. Atari is a trademark of Atari, Inc. Commodore is a trademark of Commodore International. HP-83 and HP-85 are trademarks of Hewlett-Packard Company. TRS-80 is a trademark of Radio Shack division of Tandy Corp. DEC is a trademark of Digital Equipment Corp.



	A	B	C	D
1	number:	123.4567	interval:	.125
2				
3	positive numbers:			
4	nearest:	123.5		
5	down:	123.375		
6	up:	123.5		
7				
8	negative numbers:			
9	nearest:	123.375		
10	down:	123.25		
11	up:	123.375		
12				
13	all numbers:			
14	nearest:	123.5		
15	down:	123.375		
16	up:	123.5		
17				

FIGURE 1. *This model will round any number to any specified interval, up or down. Values are entered in B1 and D1. Expressions at B4, B5, and B6 return the rounded values for positive numbers. Here, we entered the interval as 1/8 in D1, which VisiCalc evaluated to .125. The nearest eighth to 123.4567 is 123.5; the next lower eighth is 123.375; and the next greater eighth is 123.5. Different expressions are needed to round negative numbers; they are entered in rows 9, 10, and 11, shown here shaded. The two sets of expressions are combined in B14, B15, and B16, which will round any number.*

of the number we are now giving. So we need to change the expression at B4 to take this into account. Wherever the original expression was multipled by D1, we want to now divide by D1. Change B4 to:

```
>B4:@INT(B1/D1+.5)*D1
```

Now, for example, a value of 10 for D1 rounds to the nearest ten.

You can use any interval you want, it needn't be a power of 10. If you want to round to the nearest eighth, enter 1/8 in D1. The rounded value is 123.5. A value of 1/5 in D1 rounds B4 to 123.4. If D1 is 50, B4 is 100, the nearest fifty.

If you use VisiCalc Advanced Version you will recognize the similarity of this expression to the built-in function @ROUND, which rounds to a given power of 10. You supply a number and a "precision" (which must be an integer), and the function returns the number rounded to that precision. A precision of 1 rounds to the nearest tenth; 2 rounds to the nearest hundredth; −1 rounds to the nearest ten; −2 rounds to the nearest hundred, and so on. Our expression at B4 is a bit more flexible—it can round to any precision, not just to powers of ten. But @ROUND is more convenient, being built-in, and should be used when appropriate.

Forcing Rounding Up or Down

Sometimes you don't want to round to the nearest interval, rather you want to force the rounding up or down. Rounding numbers down is more straightforward, so we'll show that first.

Because @INT strips off all digits to the right of the decimal point, it always rounds positive numbers down. In the expression at B4, the factor 0.5 kicks up the rounded value if it lies halfway or more along a given interval. Eliminating that factor will result in a forced rounding down. Enter:

```
>B5:@INT(B1/D1)*D1
```

Change B1 to 123.33 and D1 to 1/5. B5 rounds the value down to 123.2, while the nearest fifth is 123.4, as shown in B4.

Rounding values up is a little trickier. Since you use a factor of 0.5 to round to the nearest interval and no factor to force rounding down, you would expect that for rounding up the expression would be:

```
>B6:@INT(B1/D1+1)*D1
```

Enter that expression. We get the correct result; 123.33 rounded up to the next fifth is 123.4. The expression works fine—with one exception. Change the value of B1 to 123.6, which lies on a fifth. The value at B6 should be 123.6, but is 123.8, one interval too high. Because the factor is 1, the expression kicks up the result by one interval whenever the original number lies exactly on an interval.

Since the problem is that at the interval itself, @INT(B1/D1 + 1)*D1 incorrectly rounds up, we can use an @IF statement in our expression to check whether our original number is on an interval. If it is, we want it to return the original number; if it is not, we want it to return the rounded number. Edit the expression to read:

```
>B6:@IF(@INT(B1/D1)=(B1/D1),B1,
                    @INT(B1/D1+1)*D1)
```

That expression uses the fact that if two numbers are evenly divisible, the integer portion of their quotient equals the quotient itself because there is no remainder.

With these three expressions we can round any positive number to any interval, in whatever way we choose.

Rounding Negative Numbers

Sometimes you may have negative numbers that need rounding, for example when analyzing a cash flow that includes money going out. Change the value at B1 to −123.4567. The rounded values are wrong because @INT will always round values toward zero. In the case of negative numbers, this means larger (less negative) numbers, which become even larger with the added factors of .5 and 1. Our expressions are set up for when @INT makes numbers smaller. To accommodate negative numbers, we essentially take the mirror images of the above expressions, interchanging the "rounding up" expression with the "rounding down" expression and taking the negative of the original number. Enter these expressions:

```
>B9:-@INT(-B1/D1+.5)*D1
>B10:@IF(@INT(-B1/D1)=(-B1/D1),B1,
                    -@INT(-B1/D1+1)*D1)
>B11:-@INT(-B1/D1)*D1
```

These expressions will work for all negative numbers, but not for positive numbers. For all-purpose expressions, ones that are applicable for every number and every interval, combine the two sets of expressions using an @IF condition:

```
>B14:@IF(B1>=0,B4,B9)
>B15:@IF(B1>=0,B5,B10)
>B16:@IF(B1>=0,B6,B11)
```

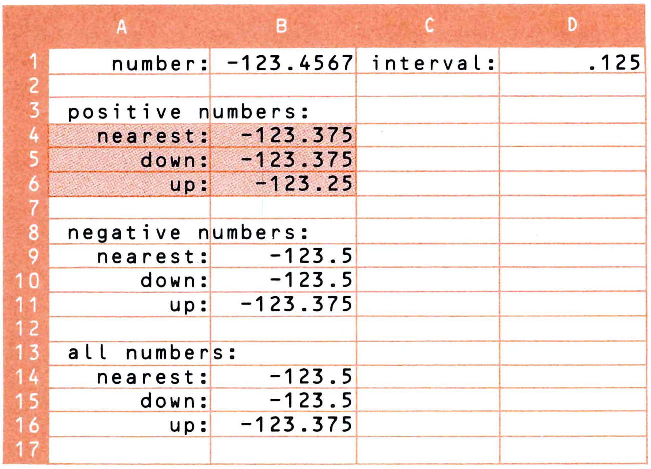
To consolidate the information, we can substitute the actual expressions for the cell references. Edit the cells to read:

```
>B14:@IF(B1>=0,@INT(B1/D1+.5)*D1,
                    -@INT(-B1/D1+.5)*D1)

>B15:@IF(B1>=0,@INT(B1/D1)*D1,@IF
      (@INT(-B1/D1)=(-B1/D1),B1,-@INT
                          (-B1/D1+1)*D1))

>B16:@IF(B1>=0,@IF(@INT(B1/D1)=
         (B1/D1),B1, @INT(B1/D1+1)*D1),
                       -@INT(-B1/D1)*D1)
```

The result is shown in Figure 2.



	A	B	C	D
1	number:	-123.4567	interval:	.125
2				
3	positive numbers:			
4	nearest:	-123.375		
5	down:	-123.375		
6	up:	-123.25		
7				
8	negative numbers:			
9	nearest:	-123.5		
10	down:	-123.5		
11	up:	-123.375		
12				
13	all numbers:			
14	nearest:	-123.5		
15	down:	-123.5		
16	up:	-123.375		
17				

FIGURE 2. *Same model as in Figure 1, illustrating rounding a negative number. Rows 4, 5, and 6 are shaded, indicating that they are not applicable for negative numbers.*

Delete rows 3 through 12 since all that information has been consolidated in the last three expressions. You can save the expressions in a file and load it whenever you need to round values, selecting the proper expression depending on whether you want to round up, down, or to the nearest interval.

For an application of rounding down, see this issue's "*Turning the Tables on Taxes* Revisited." In that article, we found the number to plug into expressions to generate a tax by rounding the taxable income down to the nearest $50.

Overcoming Value Limitations Using @LOG10

Occasionally you may need to have VisiCalc® calculate very large or very small numbers, seemingly beyond the limits of the program. In this article we show how to use @LOG10 to get around this problem. We also demonstrate how to dramatically reduce recalculation time when data is fixed. As a means of illustration, we will discuss factorials, a type of expression used in many areas of mathematics, from probability to numerical analysis.

On most machines, values must be less than 1E62 in VisiCalc (check your manual for the limit on your machine). Values greater than the limit are returned as @ERROR. To introduce the kind of problem where you might run into extremely large numbers and @ERROR, we'll use a simple example:

You are the project leader for a new product development group and plan to have your 12 engineers work in pairs. To get an idea of what your options are, you want to determine how many different pairs are possible. To calculate this, you can use factorials.

Factorials and the Problem

A factorial is a mathematical expression that is a product of positive integers such that n factorial (customarily written as "n!") is:

$$n*(n-1)*(n-2)*\ldots*3*2*1$$

For example, 3! is 3*2*1, or 6, and 4! is 4*3*2*1, or 24. By convention, 0! = 1. Notice that n! = n * (n−1)!.

Factorials get very large very quickly: 7 factorial (7!) is 5040, while 14! is 87,178,291,200.

Factorials are used frequently in statistics. You would use factorials to calculate probabilities of particular events occurring given certain conditions ("if four percent of all products are defective, what is the probability that one of the six products here is defective?") and to calculate the number of combinations of elements in a set. Our sample problem is the latter type. The solution to this type of problem is given by the equation:

$$\textit{number of combinations} = \frac{t!}{a!(t-a)!}$$

where *t* is the total number of elements and *a* is the number of objects being selected.

The solution to our problem about pairs of engineers is:

$$\textit{number of combinations} = \frac{12!}{2!(12-2)!}$$

We'll evaluate that with VisiCalc.

Set recalculation to manual:

```
/GRM
```

Enter labels flush right as shown in Figure 3. Create a list from 1 to 12 in column A by typing:

```
>A2:1
>A3:1+A2®/R:A4...A13:R
```

We'll generate a list of factorials in column B, "priming the pump" by entering 1 as the value for 1!:

```
>B2:1
>B3:+A3*B2®/R:B4...B13:RR
```

The expression in B3 uses the fact that $n! = n * (n-1)!$.

We now have a @LOOKUP table of factorials from 1! to 12!. We'll enter the total number of elements in E1 and the subset in E2 and calculate the number of combinations in E3:

```
>E1:12
>E2:2
>E3:@LOOKUP(E1,A2...A13)/
                 (@LOOKUP(E2,A2...A13)*
              @LOOKUP(E1-E2,A2...A13))
```

As shown in Figure 3, there are 66 possible combinations of 2 elements taken from a set of 12 elements.

While this problem was simple, if you are interested in large numbers of elements, the factorials can get quite big, perhaps beyond the limits of VisiCalc. The solution? Convert all your numbers to logarithms. We'll do an example using a larger set.

Say you want to determine the number of possible bridge hands, that is, the number of 13-card combinations from a 52-card deck. That value would be:

$$\textit{number of hands} = \frac{52!}{13!(52-13)!}$$

To solve that, we need to extend our @LOOKUP table. Replicate columns A and B to row 66, allowing for calculation up to 65!:

```
>A13:/R...B13:A14...A66:RRR
```

Go to row 50 to see the value of 49!. It has the value @ERROR. What happened? The value of 49! is beyond the limit for VisiCalc on most machines. You can see that 48! is about 1.24E61. That would make 49! about 49 * 1.24E61, or 6E62, too large for VisiCalc.

To solve our problem we need 52! as an intermediate value in our calculation. We don't need to display the value, but we need to have VisiCalc calculate it. Here's where logarithms come in.

Using Logarithms

Logarithms are exponents: The log of a number is the exponent to which a base is raised to obtain that number. The base is usually 10, for common logarithms, or *e*, for natural logarithms.

If:

$$a = 10^b$$

then

$$b = \log_{10} a$$

For example, 2 is the log in base 10 of 100, because $10^2 = 100$. Logs of numbers greater than 1 are considerably smaller than the numbers themselves, so logs make a nice device for working with very large numbers. We will use @LOG10, VisiCalc's function that finds logarithms in base 10, though we could just as easily use @LN, which finds natural logarithms, in base *e*.

We'll use two rules for manipulating logarithms in solving our problem.

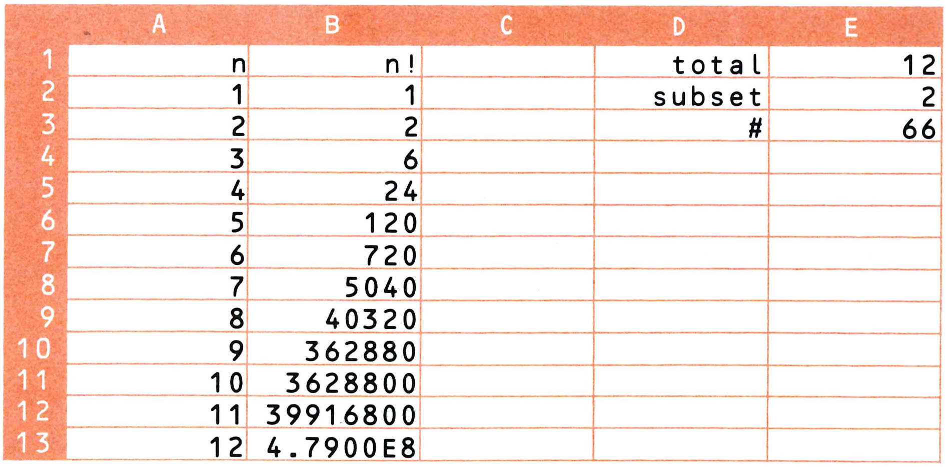
Rule 1. The log of a product is the sum of the logs of each factor:

$$\log(a * b) = \log(a) + \log(b)$$

Rule 2. The log of a quotient is the difference between the logs of each factor:

$$\log(a/b) = \log(a) - \log(b)$$

We'll make a list of logarithms of factorials in column C that we'll use to solve our bridge-hand problem. We can't simply take the logs of the values in column B as >C2:@LOG10(B2)®/R:C3 . . . C66:R because we would still have @ERROR as the values beyond 48!.



	A	B	C	D	E
1	n	n!		total	12
2	1	1		subset	2
3	2	2		#	66
4	3	6			
5	4	24			
6	5	120			
7	6	720			
8	7	5040			
9	8	40320			
10	9	362880			
11	10	3628800			
12	11	39916800			
13	12	4.7900E8			

FIGURE 3. *A simple problem involving factorials is solved using an @LOOKUP table.*

We need to use the expression that generates factorials:

$n! = n * (n-1)!$

Since the log of a product is simply the sum of the logs:

$log(n!) = log(n) + log((n-1)!)$

If we prime the pump with the first log, VisiCalc will do the rest. Since 1! equals 1, enter:

```
>C2:@LOG10(1)
```

C3 (log(2!)) equals log(2)+log(1!). Log(2) is @LOG10(A3) and log(1!) is C2. Therefore:

```
>C3:@LOG10(A3)+C2
```

Replicate this value:

```
>C3:/R:C4...C66:RR
```

We now have a list of logarithms of factorials. Figure 4 clearly shows the purpose of using logs to stay within value limits.

To use this new list in our @LOOKUP table, delete the second column:

```
>B1:/DC
```

We now have all the numbers we'll need.

Changing Expressions into Values

Our factorials are all expressions that have to be evaluated each time we recalculate the sheet. That takes a lot of time, particularly since the expressions involve logs. But these numbers are final; we don't want to change them. It is sufficient to have them simply as values. We'll change the expressions into values by saving columns A and B as a DIF file (use a disk drive specifier as appropriate):

```
>A1:/S#Slogs.dif®B66®®
```

The contents of the cells are now saved as values in *logs.dif*.

Without moving the cursor, load the file you just created over the expressions:

```
>A1:/S#Llogs.dif®®
```

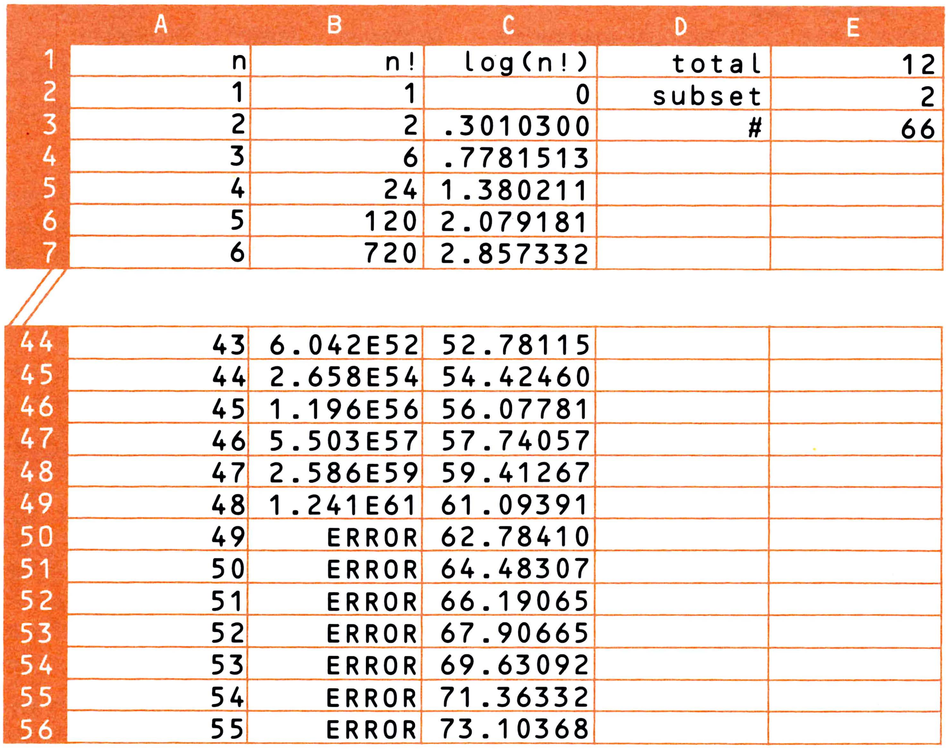
Now there are only values in columns A and B. Since they don't have to be evaluated, recalculation time is cut dramatically.

Solving the Problem

We're about ready to calculate the number of possible bridge hands.

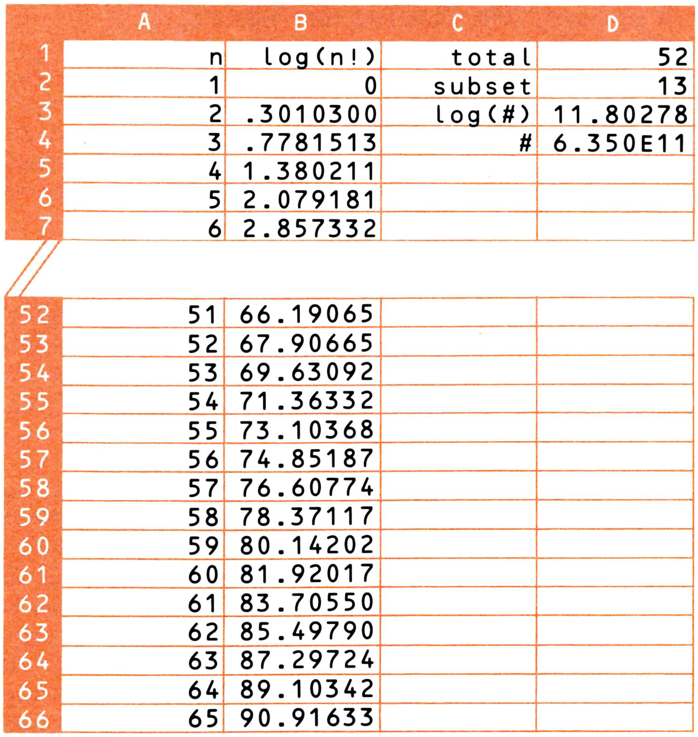
Remember that the equation is:

$$\text{number of combinations} = \frac{t!}{a!(t-a)!}$$



	A	B	C	D	E
1	n	n!	log(n!)	total	12
2	1	1	0	subset	2
3	2	2	.3010300	#	66
4	3	6	.7781513		
5	4	24	1.380211		
6	5	120	2.079181		
7	6	720	2.857332		
44	43	6.042E52	52.78115		
45	44	2.658E54	54.42460		
46	45	1.196E56	56.07781		
47	46	5.503E57	57.74057		
48	47	2.586E59	59.41267		
49	48	1.241E61	61.09391		
50	49	ERROR	62.78410		
51	50	ERROR	64.48307		
52	51	ERROR	66.19065		
53	52	ERROR	67.90665		
54	53	ERROR	69.63092		
55	54	ERROR	71.36332		
56	55	ERROR	73.10368		

FIGURE 4. *Numbers greater than about 48! cannot be calculated directly with VisiCalc (as shown in column B). Using @LOG10 in calculating factorials (column C), we were able to keep values within the limits of the program.*



	A	B	C	D
1	n	log(n!)	total	52
2	1	0	subset	13
3	2	.3010300	log(#)	11.80278
4	3	.7781513	#	6.350E11
5	4	1.380211		
6	5	2.079181		
7	6	2.857332		
52	51	66.19065		
53	52	67.90665		
54	53	69.63092		
55	54	71.36332		
56	55	73.10368		
57	56	74.85187		
58	57	76.60774		
59	58	78.37117		
60	59	80.14202		
61	60	81.92017		
62	61	83.70550		
63	62	85.49790		
64	63	87.29724		
65	64	89.10342		
66	65	90.91633		

FIGURE 5. *While the answer to our problem about the number of possible bridge hands is a value that VisiCalc can express directly (D4), its calculation involved numbers beyond the limits of the program. By using logarithms, we were able to overcome this obstacle and solve the problem.*

We are now working with logarithms so we need to modify the expression. Taking the logarithm of both sides and applying Rule 2 above, we get:

log(number of combinations) =
log(t!) – (log(a!) + log((t–a)!))

Edit the expression at D3 so that it becomes:

```
>D3:@LOOKUP(D1,A2...A66)-(@LOOKUP(D2,
   A2...A66)+@LOOKUP(D1-D2,A2...A66))
```

Be sure you modify the @LOOKUP range to go through row 66. The value in D3 is now the log of the number of combinations. Change the label at C3 to reflect this. At D4 we'll convert back to the number of combinations:

```
>D4:10^D3
```

(If you had used @LN instead of @LOG10 to modify the numbers, D4 would read @EXP(D3).)

Enter the numbers for our bridge-hand problem:

```
>D1:52
>D2:13
```

Now recalculate. As shown in Figure 5, there are 6.35 x 10^{11} possible bridge hands. That's 635,000,000,000. Your chance of being dealt an entire suit in a bridge hand is about one in 160 billion.

By converting everything into logs, performing the calculation, and converting back, we were able to stay within the bounds allowed by VisiCalc. This strategy also works with extremely small numbers; VisiCalc's lower limit on most machines is 1E–66.

Factorials are just one example of where the log trick is handy. A chemist working with molecules or an astronomer working with galactic distances could require numbers too small or too big to handle conventionally with VisiCalc. Using @LOG10 or @LN, the problems easily can be cut down to size.

Book Review

Reviews published in SATN are the opinions of the reviewers, not of Software Arts, and may not be construed as official positions or endorsements by Software Arts.

Doing Business With VisiCalc

by Stanley R. Trost. Sybex U.S.A., Berkeley, 200 pp.
Reviewed by Ronald Klinger

Doing Business With VisiCalc will be helpful to a beginner, but most useful for the VisiCalc® user who is already fairly familiar with the VisiCalc manual, the intermediate user, if we may call him that.

The Trost book contains a good, diverse mix of the kinds of business problems one may solve with VisiCalc. Of the more than 40 of them, most are relatively simple. But the ideas behind their construction are what count and those ideas come through clearly, providing the reader with sufficient information to construct more intricate and sophisticated spreadsheets. Both the diversity of business and investment applications and the author's exposition are the outstanding contributions of the work.

A perhaps petty complaint: I found the author's tutorial style to be excessively simple. Although not aimed at the advanced VisiCalc user, the average reader of this volume might be sophisticated in business, and able to learn quickly. Yet, for example, throughout the book, most spreadsheet instructions begin with a /CY, clearing the screen, and setting certain default parameters. One would think that those things could be dispensed with, for most reasonably intelligent readers, early in the game.

On the more positive side, Trost has built a series of spreadsheets for record-keeping, finance, budgeting, sales, manufacturing, real estate, and income taxes, and organized the book around them. Taken together they utilize a wide assortment of VisiCalc commands. For example, in the chapter on record-keeping, which appears early in the book, Trost shows how to use the DIF™ format to create an annual income statement consisting in part of monthly expense registers. I prefer Trost's explanation of the DIF format to the discussion in the manual. It's less technical. But more important, it makes its point by using basic examples. The reader can quickly see the impact and advantages of transferring data from one file or program to another.

Perhaps because of the time it takes to publish a book, some of the examples are out of date. In the chapter on finance, for example, is a spreadsheet on depreciation methods. The methods shown, while helpful in learning VisiCalc, are unfortunately no longer allowed under the law if the property being depreciated was placed in service after 1980. The new method, called the "Accelerated Cost Recovery System," or ACRS, is mandatory. To use the ACRS method with VisiCalc, one should set up tables with the appropriate percentages and then use the @LOOKUP technique that Trost describes in examples on pages 38 and 229. (See also *ACRS Depreciation*, SATN, Vol. II, No. 1, pp. 6-8 and Vol. II, No. 5, pp. 2-5.)

Other spreadsheets dealing with depreciation, for example the condominium and office investment examples, are similarly outdated.

When one has gone through the volume and actually sat down at the computer and carried out the many examples, the ability to follow through with additional and sophisticated techniques is bound to follow. The value of the book to the very experienced VisiCalc user is limited. There are no unusual or "inspiring" applications (if we may think of VisiCalc as inspirational).

In the chapter on sales, the author uses VisiPlot® to graph data contained on a couple of spreadsheets. Although the models chosen were elementary, the capabilities of combining VisiCalc and VisiPlot were aptly demonstrated.

There is a 16-page Appendix containing short descriptions of the standard VisiCalc commands and conventions, a handy addition to anyone needing a quick reference to jog the memory. The book appeared at about the same time as the Advanced Version of the VisiCalc program, allowing the publisher and author to include only a two-page summary of a few of the features of the Advanced Version.

In summary, the book is a superior tutorial for those VisiCalc users who have read the manual and are ready to adapt the software to a variety of business and investment uses. I highly recommend it.

Ronald Klinger, the editor of The Journal of Taxation, *is an attorney and tax practitioner doing tax and financial consulting work in New York.*

BACK ISSUES AVAILABLE

Back issues of SATN are now available. The cost is $6.00 per copy for orders mailed within the USA. If your order is to be mailed outside the USA, add $2.50 for shipping and handling to the full price of the order.

To order back issues, please send your name and address, a list of the specific issues of SATN you want (by Volume and Issue numbers), and a check or money order payable to Software Arts Products Corp. to:

SATN Back Issues
27 Mica Lane
Wellesley, MA 02181

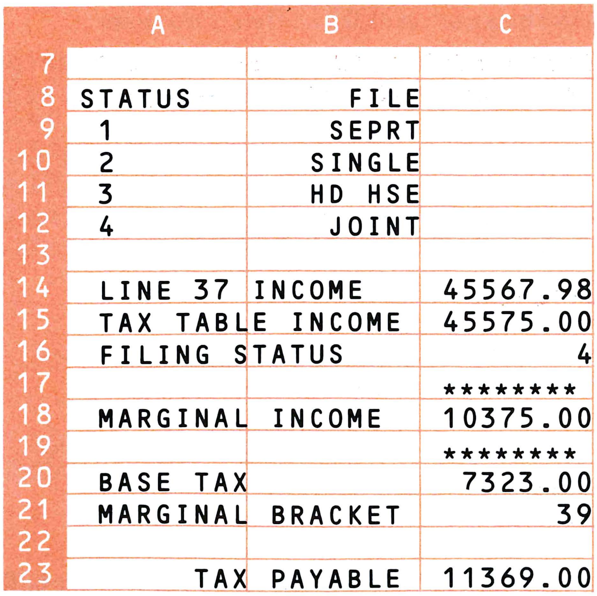
Turning the Tables on Taxes Revisited

In *Turning the Tables on Taxes* (March/April 1983), we showed how to rewrite the Federal Tax Rate Schedules (Schedules X, Y, and Z) to use them with the VisiCalc® program. As the source for our model, we used the schedules shown in a publication we had received in the Fall of 1982 from an accounting firm.

As Barry Bayer (a reader and sometime contributor to SATN) pointed out, however, Schedules X, Y, and Z are only applicable if taxable income is more than $50,000 or if Schedule G is used for income averaging. If taxable income is less than $50,000, taxpayers have to use the 1982 Tax Table, found on pages 34-39 of the Instructions for preparing Form 1040.

Consequently, we were afraid that the model we set up had only marginal utility, since only a few of the base tax amounts and marginal tax rates shown in the Rate Schedules are for incomes in excess of $50,000. It turns out, however, that essentially all the tax values in the six-page Tax Table are derived from the one-page Tax Rate Schedules, the values of which are already in our model. Because of this connection, we can easily expand our model using the rounding technique described in *Round and Round We Go*, regardless of whether the Tax Table or the Tax Rate Schedules should be used.

The Tax Table breaks down income into more than 950 intervals and specifies a single tax for any income that is within an interval. At incomes above $3000, the intervals are $50. Thus, an unmarried taxpayer with a taxable income anywhere from $15,500 to $15,550 pays a tax of $2472. The Tax Rate Schedules, on the other hand, treat income more continuously, breaking it into about a dozen intervals of varying sizes. Within each interval, the tax is computed using a base tax and a marginal tax rate. Using the appropriate Rate Sched-



```
          A         B         C
 7
 8  STATUS            FILE
 9    1              SEPRT
10    2             SINGLE
11    3             HD HSE
12    4              JOINT
13
14    LINE 37  INCOME     45567.98
15    TAX TABLE INCOME    45575.00
16    FILING STATUS              4
17                        ********
18    MARGINAL INCOME     10375.00
19                        ********
20    BASE TAX             7323.00
21    MARGINAL BRACKET          39
22
23         TAX PAYABLE    11369.00
```

FIGURE 6. *Compare this figure with Figure 3 in the March/April issue of SATN.*

If line 37 (taxable income) is—		And you are—			
At least	But less than	Single	Married filing jointly	Married filing separately	Head of a household
		Your tax is—			
45,000					
45,000	**45,050**	13,831	11,155	16,237	12,657
45,050	**45,100**	13,856	11,174	16,262	12,682
45,100	**45,150**	13,881	11,194	16,287	12,706
45,150	**45,200**	13,906	11,213	16,312	12,731
45,200	**45,250**	13,931	11,233	16,337	12,755
45,250	**45,300**	13,956	11,252	16,362	12,780
45,300	**45,350**	13,981	11,272	16,387	12,804
45,350	**45,400**	14,006	11,291	16,412	12,829
45,400	**45,450**	14,031	11,311	16,437	12,853
45,450	**45,500**	14,056	11,330	16,462	12,878
45,500	**45,550**	14,081	11,350	16,487	12,902
45,550	**45,600**	14,106	11,369	16,512	12,927
45,600	**45,650**	14,131	11,389	16,537	12,951
45,650	**45,700**	14,156	11,408	16,562	12,976
45,700	**45,750**	14,181	11,428	16,587	13,000
45,750	**45,800**	14,206	11,447	16,612	13,025
45,800	**45,850**	14,231	11,468	16,637	13,049
45,850	**45,900**	14,256	11,490	16,662	13,074
45,900	**45,950**	14,281	11,512	16,687	13,098
45,950	**46,000**	14,306	11,534	16,712	13,123

Schedule Y

Married Filing Joint Returns and Qualifying Widows and Widowers

Use this schedule if you checked Filing Status Box 2 or 5 on Form 1040—

If the amount on Form 1040, line 37 is:		Enter on Form 1040, line 38	
Over—	*But not Over—*		*of the amount over—*
$0	$3,400	—0—	
3,400	5,500	...........12%	$3,400
5,500	7,600	$252+14%	5,500
7,600	11,900	546+16%	7,600
11,900	16,000	1,234+19%	11,900
16,000	20,200	2,013+22%	16,000
20,200	24,600	2,937+25%	20,200
24,600	29,900	4,037+29%	24,600
29,900	35,200	5,574+33%	29,900
35,200	45,800	7,323+39%	35,200
45,800	60,000	11,457+44%	45,800
60,000	85,600	17,705+49%	60,000
85,600	....	30,249+50%	85,600

FIGURE 7. *To produce the shaded column in the Tax Table printed above, you need only round the amount shown on Line 37 of Form 1040 down to the nearest $50, add $25, and then use the appropriate values, printed in rust, from Schedule Y.*

ule, an unmarried taxpayer with a taxable income of $15,500 would pay $2465; with an income of $15,550, a tax of $2479.

Notice that the tax derived from the Table is midway between the taxes derived from the Rate Schedules for the incomes that make up the boundaries of the Table's interval. That relationship, true in all cases, allows us to use the expressions that generated the tax using the Rate Schedules to also generate the tax when the Tax Table has to be used. All we have to add to the model we developed for the original article is a calculation that determines which $50 interval a taxpayer's taxable income falls into in the Table, go to the midpoint of that interval, and use that figure in our expressions. In other words, we have only to round taxable income *down* to the next $50, add $25, and use the result.

Using the rounding technique developed in *Round and Round We Go*, the expression would be:

*@INT(taxable income/50) *50 + 25*

We'll enter this expression in the model we developed in the March/April 1983 issue (refer to page 6).

On the sheet shown in the original article we inserted a row at C15 to create Figure 6. We then relabelled row 14 as "LINE 37 INCOME," referring to the line on Form 1040 on which Taxable Income is entered. As shown in the figure, that is entered as a value at C14. Then, at C15 we entered the calculation that finds the midpoint of an income interval. Enter:

```
>C15:@INT(C14/50)*50+25
```

C15 now will be the value rounded to the midpoint of the interval specified in the Tax Table. Since the Tax Table is only applicable if taxable income is less than $50,000, we have to test for that condition. Edit the expression at C15 to read:

```
>C15:@IF(C14<50000,
                    @INT(C14/50)*50+25,C14)
```

That is, if the taxable income is less than $50,000, return the rounded number; if not, return the taxable income itself.

The IRS, when deriving the Tax Table from the Tax Rate Schedules, appears to have done some additional rounding. When the rate of an individual's marginal bracket is applied to his or her marginal income it sometimes produces a tax that is not a pure integer. The IRS has rounded such calculations to the nearest dollar. Thus the calculation of TAX PAYABLE at C23 (C22 on the original sheet) should be edited to read:

```
>C23:@MAX(@INT(C21/100*C18+.5)+C20,0)
```

Now your model will work for all incomes. When next year's tax tables come out, all you need do is change the numbers for base tax and marginal rate and you'll be in business.

(Since the Rate Schedules are also used for income averaging, you could also build in a flag in the expression at C15 to check whether income averaging is being used—if it is, use the taxable income as is, regardless of its value.)

One caveat: Below a taxable income of $3000, the Tax Table uses $25 intervals. Our model assumes $50 intervals so the tax amount returned may be off by a dollar or two below $3000.

RELATED PRODUCTS CHECKLIST

The information contained in this column has been garnered from a variety of sources including dealers, press releases, manufacturers, and advertising.

No one at Software Arts has reviewed or examined any of these products to determine whether they perform as claimed or as listed below. An inclusion in this listing does not constitute an endorsement by Software Arts of that product. We also wish to point out that if any of the products listed alter, patch, or copy the VisiCalc program, they could be in violation of your user licensing agreement and/or the copyright.

Since we have not reviewed any of the products, we cannot provide you with any more information than is contained in the list. If you want to learn more or you wish to purchase any of the products listed, contact your dealer first. Although we have identified distributors in some listings, many distributors do not accept orders from individuals for single copies of software or pieces of hardware.

SOFTWARE

Only software that claims to complement the VisiCalc program will be listed.

Communications

Apple-IBM Connection

- Runs on Apple II and IBM PC

Allows transfer of VisiCalc models and files developed on the Apple II to the IBM PC without reentering data from the keyboard. Can also send electronic mail messages between the two systems.

Distributed by Alpha Software Corp., Burlington, MA.

Suggested retail price: $195.

Multi-Calc

- Runs on Apple II

Enables Cluster/One system users to exchange VisiCalc worksheets with those of other network users. Makes standard services of the Nestar network available to VisiCalc users, including spooled printing, large-capacity disk storage, electronic mail, and local or remote transferring of files.

Distributed by Nestar, Palo Alto, CA.

Suggested retail price: $95.

RVC (Reflexive VisiCalc)

- Runs on Apple II

Allows two users of VisiCalc to interact on the same spreadsheet over a telephone line.

Distributed by Arrow Micro Software, Kanata, Ontario, Canada.

Suggested retail price: $45.

VIS/Bridge/DJ

- Runs on Apple II+, Apple ///, IBM PC, and Radio Shack TRS-80 Models I, II, III, 16

Loads investment data from Dow Jones News/Retrieval into VisiCalc. Users who do not utilize Dow Jones News/Retrieval service can still use VIS/Bridge/DJ but must sign up with Dow Jones.

Distributed by Solutions, Inc., Montpelier, VT.

Suggested retail price: $445.

VIS/Bridge/GL

- Runs on Radio Shack TRS-80 Model II

Retrieves data from Tandy Corporation's General Ledger Package for the TRS-80 Model II directly to the VisiCalc spreadsheet.

Distributed by Solutions, Inc., Montpelier, VT.

Suggested retail price: $195.

VisiTran/DATA TRANSLATOR

- Runs on Apple II
- Supports DIF format

Utility program that allows transfer of data from AppleSoft BASIC programs into VisiCalc.

Distributed by ADC Associates, Palo Alto, CA.

Suggested retail price: $99.

Templates

Accounting Plus II, Payroll Plus, and Plus Package

- Run on Apple II and Apple /// in emulation mode

Three accounting packages that, with the product Data Plus, work with VisiCalc. Accounting Plus II consists of four modules: general ledger, accounts payable, accounts receivable, financial reports and statements. Suggested retail price: $395 per module, $1,250 for all four.

Payroll Plus maintains payroll information, generates payroll checks and reports, and allows job costing. Can be used with Accounting Plus II. Suggested retail price: $425.

Plus Package consists of four parts: General Ledger Plus, Invoices Plus, Labels Plus and Data Plus, which converts data for use with VisiCalc. Suggested retail price: $99-$295.

Distributed by: Software Dimensions, Citrus Heights, CA.

SofTax

- Runs on Apple II, Apple ///, and IBM PC

A tax analysis, simulation, and preparation package, designed for individuals, partnerships, and corporations. Menu-driven format. Prepares final returns.

Distributed by Design Trends, Wilton, CT.

Suggested retail price: $395.

Training

Cdex Training for VisiCalc

- Available for Apple II+ and IBM PC

Includes seven worksheets: a P&L statement, a pricing model, a personal net worth statement and four more. It adds nine exercises for more advanced features and puts a command reference guide in the front of the manual.

Distributed by Cdex Corporation, Los Altos, CA.

Suggested retail price: $49.95.

How to Get Started on VisiCalc

Introduces the commands and features of VisiCalc as you build your own cash flow model. Uses three spoken-voice cassettes.

Distributed by: FlipTrack Systems, Glenn Ellyn, IL.

Suggested retail price: $49.95.

VisiCalc Program Made Simple Learning System Tutorial

- For users of the IBM PC

An interactive instructional video cassette tape and a "VideoGuide" reference booklet designed to teach the individual, small business, and corporate microcomputer user how to set up and use the VisiCalc program.

Distributed by: MicroVideo Learning Systems, New York, NY.

Suggested retail price: $99.95.

The VisiCalc Training Videotape

An application-oriented videotape for the data processing, management, and legal professions. Conducted by Sanford Sorkin, a computer seminar course leader for the American Management Associations.

Distributed by Legalvision Inc., New York, NY.

Suggested retail prices: VHS, $195; Beta, $195; ¾", $235.

INDEX TO VOLUME II

[AV] refers to specific mentions of functions or commands in the VisiCalc Advanced Version program.

^< p.12 #4 [AV]
^> p.12 #4 [AV]
^R p.12 #4 [AV]
.PRF p.2 #3;p.9 #4
/A(Attributes) p.11 #1 [AV]
/AHN p.12 #4 [AV]
/AV% p.11 #1 [AV]
/B p.2 #4
/CS p.11 #3 [AV]
/F p.11 #3 [AV]
/GAHY p.12 #4 [AV]
/P p.1 #3;p.3 #3
/PD p.2 #3
/PF p.2 #3;p.3 #3;p.8 #4
/R p.3 #2;p.2 #4;p.12 #4 [AV]
/R(A) p.12 #4 [AV]
/SS p.1 #3;p.4 #4
/S#L p.4 #4;p.6 #6
/S#S p.2 #1;p.4 #4;p.6 #6
@ABS p.9 #2
@AND Review of Syntax p.8 #1
@AVERAGE p.3 #2
@CHOOSE p.7 #1
  @CHOOSE and Boolean Values p.8 #1
@COUNT p.4 #2
  @COUNT and Labels p.5 #2
@DOTPROD p.10 #2 [AV]
@ERROR p.3 #5;p.4 #6
@EXP p.9 #2
@FALSE p.8 #1 (See Boolean Values)
@IF p.11 #1
  Nesting @IF within @IF p.2 #5
  Review of Syntax p.8 #1
@INT p.2 #4;p.2 #6;p.9 #6
@LABEL p.11 #1 [AV]
@LCHOOSE p.11 #1 [AV]
@LOG10 p.4 #6
@LOOKUP p.2 #4
@MAX p.6 #4
@MIN p.7 #3
@NA p.10 #1
@OR p.3 #5
@PI p.9 #2
@ROUND p.3 #6 [AV]
@SQRT p.6 #2;p.8 #2
@SUM p.9 #2;p.7 #3
  @SUM and Labels p.5 #2
  Including other functions within @SUM p.7 #3
@TRUE p.8 #1 (See Boolean Values)
Accelerated Cost Recovery System (ACRS)
  Personal Property p.6 #1
  Real Property p.2 #5
ASCII Text File p.9 #4
Books Reviewed
  Anbarlian, *An Introduction to VisiCalc Matrixing for Apple and IBM* p.6 #2
  Beil, *The VisiCalc Book, Apple Edition* p.5 #3
  Desautels, *VisiCalc for the IBM Computer* p.7 #4
  Simpson, *VisiCalc Programming: No Experience Necessary* p.5 #5
  Trost, *Doing Business with VisiCalc* p.7 #6

Boolean Values p.8 #1
Chi-Square p.7 #2
Coefficient of Correlation p.5 #2
DIF p.2 #1;p.4 #4;p.6 #6
Displaying Expressions p.11 #2 [AV]
EasyWriter II p.10 #5
Future Value
  Of a Series of Equal Deposits p.7 #5
  Of a Single Deposit p.7#5
General Expressions
  Chi-Square p.9 #2
  Coefficient of Correlation p.6 #2
  Future Value of a Series of Equal Deposits p.7 #5
  Future Value of a Single Deposit p.7 #5
  Intercept of a Line p.5 #2
  Loan Repayment p.5 #1;p.7 #5
  Present Value of a Series of Equal Deposits p.7 #5
  Present Value of a Single Deposit p.7 #5
  Rule of 78 p.5 #1
  Sinking Fund p.7 #5
  Slope of a Line p.4 #2
  Standard Deviation p.8 #2
  Sum of the Years' Digits Depreciation p.5 #1
Intercept of a Line p.5 #2
Keystroke Memory p.12 #4 [AV]
Least Squares Fit p.4 #2
Loan Repayment p.7 #5;p.5 #1
Magic Window p.1 #3
Mean and Standard Deviation p.7 #2
Normal Distribution (or Gaussian) Curve p.8 #2;p.9 #2
Present Value
  Of a Series of Equal Deposits p.7 #5
  Of a Single Deposit p.7 #5
Print File p.2 #3
Print to Disk p.2 #3
Quarterly Ratio Index p.3 #2
Rule of 78 p.5 #1
Sinking Fund p.7 #5
Slope of a Line p.4 #2
Standard Deviation p.8 #2
Sum of the Years' Digits Depreciation p.5 #1
Tax Rate Schedules for Individuals p.2 #4
User Defined Formats p.11 #3 [AV]
VisiCalc Advanced Version p.11 #1;p.10 #2;p.12 #4
VisiCalc Advanced Version Commands and Functions
  ^< p.12 #4
  ^> p.12 #4
  ^R p.12 #4
  /A (Attributes) p.11 #1
  /AHN p.12 #4
  /AV% p.11 #1
  /CS p.11 #3
  /F p.11 #3
  /GAHY p.12 #4
  /R p.11 #2;p.12 #4
  /R(A) p.12 #4
  @DOTPROD p.10 #2
  @LABEL p.11 #1
  @LCHOOSE p.11 #1
  Displaying Expressions p.11 #2
  Keystroke Memory p.12 #4
  User Defined Formats p.11 #3
  VisiTrend/VisiPlot p.1 #1
  Wordstar p.8 #4

IN OUR NEXT ISSUE

VisiCalc and Screenwriter II
Craig Stinson resumes his series on using VisiCalc with word processing programs. In the September/October issue, he examines Sierra On-Line Systems' popular word processor for the Apple.

Curve Fitting with VisiCalc
Sometimes it's clear that the numbers you're dealing with are going up or down, but it's not so clear whether the trend is a nice, straight line. We'll look at how to use the VisiCalc program to find regression lines that describe a set of data points, select the most likely one, and then do some forecasting of projected changes.

Executive Computing
One of the early books about using microcomputers in business, *Executive Computing,* was written by John Nevison and appeared just in time to include an appendix describing a new-to-the-market program called VisiCalc. Nevison's book has been updated, with VisiCalc used extensively, and republished as a disk and manual. We will look at it and at Edouard Desautels' *Advanced Uses of VisiCalc,* another disk and manual set, in the September/October issue.